**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра «Информационные системы»**

Реферат №2

**по дисциплине «Системы реального времени»**

Тема: **Диспетчеризация задач**

|  |  |  |
| --- | --- | --- |
| Студентгр. 9373 |  | Блохина А.А. |
| Преподаватель |  | Сидельников В.В. |

Санкт-Петербург

2023

**содержание**

[1. Задачи - процессы и потоки 3](#_Toc128742127)

[2. Вытеснение, time-slicing 4](#_Toc128742128)

[3. Диспетчеризация задач, определение 5](#_Toc128742129)

[4. Состояния задачи 5](#_Toc128742130)

[5. События, вызывающие смену состояний 6](#_Toc128742131)

[6. Очереди задач 7](#_Toc128742132)

[7. Приоритетное управление 8](#_Toc128742133)

[8. Статические и динамические алгоритмы 9](#_Toc128742134)

[Список использованной литературы: 10](#_Toc128742135)

# Задачи - процессы и потоки

Задачи, процессы и потоки - это основные понятия, связанные с управлением вычислительными ресурсами в операционных системах. Каждое из этих понятий относится к выполнению программного кода на компьютере, но имеет свои уникальные особенности.

Задачи (Tasks) - это базовые единицы работы в операционной системе. Задача представляет собой единственный поток выполнения, который может выполняться на процессоре. Задача может быть запущена, приостановлена, возобновлена и завершена операционной системой. Внутри задачи может быть много потоков, которые выполняются параллельно.

Процесс (Processes) - это абстракция, реализованная на программном уровне - уровне операционной системы. Процесс был придуман для организации всех данных, необходимых для работы программы. Можно сказать, что процесс - это просто контейнер, в котором находятся ресурсы программы.

Элементы процесса:

* адресное пространство
* потоки
* открытые файлы
* дочерние процессы

Поток (Threads) - это сущность, в которой выполняются процедуры программы. Поток легче, чем процесс, и создание потока стоит дешевле. Потоки используют адресное пространство процесса, которому они принадлежат, поэтому потоки внутри одного процесса могут обмениваться данными и взаимодействовать с другими потоками.

Элементы потока:

* счётчик команд
* регистры
* стек

Каждый из этих концептов имеет свои уникальные преимущества и недостатки, и выбор того, какой из них использовать, зависит от конкретной задачи, которую нужно решить. Задачи обычно используются для организации высокоуровневых процессов, которые могут быть приостановлены или завершены в любой момент. Процессы используются для организации отдельных приложений, которые могут работать независимо друг от друга. Потоки используются для управления многопоточными приложениями, которые могут параллельно обрабатывать большое количество данных.

# Вытеснение, time-slicing

Вытеснение (preemption) и time-slicing - это механизмы, используемые операционными системами для управления доступом процессов и потоков к ресурсам процессора.

Вытеснение - это механизм, при котором операционная система принудительно приостанавливает выполнение текущего процесса или потока и переключается на выполнение другого процесса или потока. Процессы и потоки могут быть вытеснены, когда они ждут завершения операции ввода-вывода, когда они блокируются на ожидании событий, когда процессор нужен для выполнения более важной задачи или когда время, выделенное на выполнение процесса или потока, истекает. Вытеснение может происходить как внутри одного процесса, так и между разными процессами.

Time-slicing - это механизм, при котором процессор разделяется между несколькими процессами или потоками, предоставляя каждому из них определенный период времени для выполнения своей работы. В конце каждого периода времени процессор переключается на выполнение другого процесса или потока, независимо от того, закончен ли текущий процесс или поток. Time-slicing позволяет обеспечивать справедливое распределение времени процессора между различными процессами и потоками и избегать ситуации, когда один процесс или поток занимает процессор на неопределенное время, блокируя другие процессы и потоки.

# Диспетчеризация задач, определение

Диспетчеризация задач - это процесс управления и распределения ресурсов процессора между различными процессами и потоками в операционной системе. Диспетчер является частью ядра операционной системы и отвечает за выбор процесса или потока, который будет запущен и выполнен следующим на процессоре.

# Состояния задачи

Состояния задачи - это различные стадии жизненного цикла процесса в операционной системе, которые определяют текущее состояние выполнения задачи. Обычно каждый процесс может находиться в одном из нескольких возможных состояний, в зависимости от того, что происходит с процессом в данный момент времени.

Ниже перечислены наиболее распространенные состояния задачи:

* Готовый (Ready) - процесс готов к выполнению, но еще не запущен на процессоре. Процесс находится в очереди готовности и ожидает своей очереди на выполнение.
* Запущенный (Running) - процесс выполняется на процессоре. Это означает, что процессор выделяет ресурсы для выполнения процесса, который находится в данном состоянии.
* Блокированный (Blocked) - процесс не может продолжать выполнение, так как он ожидает на выполнение некоторой операции, которая может быть выполнена только другим процессом или системой ввода-вывода. В этом состоянии процесс не выполняется и находится в ожидании выполнения нужной операции.
* Приостановленный (Suspended) - процесс временно остановлен операционной системой. В этом состоянии процесс не выполняется и может быть возобновлен позже. При этом ресурсы, занятые процессом, могут быть освобождены для других процессов.
* Завершенный (Terminated) - процесс завершил свое выполнение и был удален из системы.

# События, вызывающие смену состояний

Смена состояний задачи в операционной системе происходит в ответ на различные события. Ниже приведены наиболее распространенные события, которые могут вызвать смену состояния задачи:

* Создание задачи (Creation) - когда процесс создается и начинает свое выполнение, он переходит из состояния "новый" в состояние "готовый".
* Ожидание события ввода-вывода (I/O wait) - когда процесс ожидает завершения операции ввода-вывода, он переходит в состояние "блокированный".
* Завершение операции ввода-вывода (I/O completion) - когда операция ввода-вывода завершена, процесс переходит из состояния "блокированный" в состояние "готовый".
* Закончилось время выделенное на выполнение (Time slice expired) - когда процесс превышает максимальное время, которое было выделено на его выполнение, операционная система приостанавливает выполнение процесса и переводит его в состояние "готовый".
* Ожидание события синхронизации (Synchronization wait) - когда процесс ожидает наступления какого-то события, например, завершения другого процесса, он переходит в состояние "блокированный".
* Событие завершения (Termination event) - когда процесс завершает свое выполнение, он переходит в состояние "завершенный".

В операционных системах очередь задач - это механизм, который используется для хранения задач, ожидающих выполнения. Когда задача готова к выполнению, она становится частью очереди и ждет своей очереди на выполнение.

# Очереди задач

Очереди задач в операционных системах могут быть реализованы на разных уровнях, их можно классифицировать по различным критериям. Однако, наиболее распространенной классификацией является классификация очередей задач по приоритету.

Существует несколько типов очередей задач, которые могут использоваться в операционных системах:

Очереди задач с фиксированным приоритетом (Fixed-priority scheduling queues) - задачи помещаются в очередь в порядке возрастания приоритета. Задачи с более высоким приоритетом будут выполняться раньше, чем задачи с более низким приоритетом.

Очереди задач с приоритетом, который изменяется динамически (Dynamic-priority scheduling queues) - приоритет задачи может изменяться в зависимости от ее поведения. Например, если задача выполняется дольше, чем ожидалось, ее приоритет может быть увеличен, чтобы дать ей больше времени на выполнение.

Очереди задач с круговым расписанием (Round-robin scheduling queues) - каждая задача получает равную порцию времени на выполнение. Когда процесс выходит из состояния выполнения, он становится в конец очереди и ожидает своей очереди на выполнение.

Очереди задач с приоритетным круговым расписанием (Priority-based round-robin scheduling queues) - задачи разбиваются на группы по приоритету, каждая группа получает свой круговой расписание.

# Приоритетное управление

Приоритетное управление - это способ управления процессами в операционной системе, при котором каждому процессу или потоку назначается определенный приоритет, и процессы с более высоким приоритетом получают больше ресурсов и обрабатываются быстрее, чем процессы с более низким приоритетом.

Приоритет процесса или потока может быть задан явно программистом или может быть автоматически определен операционной системой, и может зависеть от различных факторов, таких как уровень важности процесса, использование ресурсов и т.д.

При приоритетном управлении, операционная система использует планировщик процессов или планировщик потоков для принятия решения о том, какой процесс или поток должен быть запущен и какой должен ожидать. Планировщик использует приоритеты процессов и потоков, а также различные алгоритмы для определения того, какой процесс или поток следует обрабатывать следующим.

Приоритетное управление может помочь увеличить производительность и эффективность операционной системы, позволяя быстрее обрабатывать задачи, которые считаются более важными или критическими, и уменьшать время ожидания для процессов с более высоким приоритетом. Однако слишком большое количество процессов с высоким приоритетом может привести к неравномерному распределению ресурсов и ухудшению производительности для процессов с более низким приоритетом. Поэтому, оптимальное использование приоритетного управления требует баланса между различными процессами и потоками и правильного определения приоритетов в соответствии с требованиями системы.

# Статические и динамические алгоритмы

Алгоритмы распределения, использования, освобождения ресурсов и представления к ним доступа предназначены для наиболее эффективной организации работы всего комплекса устройств ЭВМ. Выделяются 2 вида алгоритмов: статистические и динамические.

Статические алгоритмы - это алгоритмы, которые разработаны для обработки фиксированного набора данных. Они используются в ситуациях, когда размер данных, на которых должен работать алгоритм, заранее известен и не изменяется в процессе выполнения программы. Эти алгоритмы обычно проще и быстрее, чем динамические алгоритмы, потому что они могут оптимизироваться для конкретных наборов данных. Однако они могут быть неэффективными в случаях, когда размер данных неизвестен заранее или может изменяться в процессе выполнения программы.

Динамические алгоритмы - это алгоритмы, которые могут изменять свое поведение в зависимости от изменений во входных данных. Они используются в ситуациях, когда размер данных неизвестен заранее или может изменяться в процессе выполнения программы. Они могут обрабатывать изменения в данных и адаптироваться к новым условиям в процессе выполнения программы. Эти алгоритмы обычно более сложны и медленнее, чем статические алгоритмы, потому что они должны учитывать все возможные изменения в данных. Однако они могут быть более эффективными в случаях, когда размер данных может изменяться или когда требуется быстрая адаптация к изменяющимся условиям.

**Список использованной литературы:**

1. Сергей Сорокин. Системы Реального Времени. // СТА. – 1997. - №2. – С 22-29.
2. IEEE Standard Glossary of Software Engineering Terminology. / New York : The Institute of Electrical and Electronics Engineers, 1990. – URL: <http://www.informatik.htw-dresden.de/~hauptman/SEI/IEEE_Standard_Glossary_of_Software_Engineering_Terminology%20.pdf>
3. Martin Timmerman, Bart Van Beneden, Lourent Uhres. RTOS Evaluation Kick Off! // Real-Time Magazine. – 1998. – N3 pp 6 – 10.
4. Сидельников В.В. Лекции по дисциплине «системы реального времени». СПб.: Изд-во СПбГЭТУ «ЛЭТИ»
5. Системное программное обеспечение управляющих систем реального времени: учебное пособие / М.В. Кавалеров. – Пермь: Изд-во ПНИПУ, 2013. – 190 с.
6. Гриценко Ю.Б. Системы реального времени: Учебное пособие. — Томск: Томский межвузовский центр дистанционного образования, 2009. — 256 с
7. Операционные системы реального времени: учебное пособие / Егор Горошко. – Харьков: Изд-во ИРЭ НАН Украины, 2003.
8. Интернет источники URL:
9. <https://tproger.ru/problems/what-is-the-difference-between-threads-and-processes/> (21.02.2023)
10. <https://en.m.wikipedia.org/wiki/Preemption_(computing)> (21.02.2023)
11. <https://studfile.net/preview/2874724/page:3/> (21.02.2023)
12. <https://studfile.net/preview/919468/#:~:text=Диспетчеризация%20процессов%20(задач)%20–%20это,готовности%20в%20состояние%20выполнения%20(счёта)> (21.02.2023)
13. <https://crm.ru/instruction/Standart/Cooperation/3281/> (21.02.2023)
14. <http://www.pic24.ru/doku.php/tnkernel/ref/task/intro> (21.02.2023)
15. <https://dev.abcdef.wiki/wiki/Process_state> (21.02.2023)
16. <https://poisk-ru.ru/s81841t1.html> (22.02.2023)
17. <https://csperkins.org/teaching/2004-2005/rtes5/lecture10.pdf> (22.02.2023)